Understanding the Growth in Data Science StackOverflow Questions

Alana Willis, Clare Cruz, Dan Nason, & Megan Christy
Department of Statistics and Data Science
Carnegie Mellon University

April 24, 2022

## Executive Summary

StackOverflow is a public question-forum that is popular among data scientists and programmers. In this analysis, it is used to quantify the use of programming languages, such as Python and R, and data science topics, such as Regression and Time Series. Using time series modeling methods on monthly question counts of such languages and topics, we hope to see which data science topics significantly contribute to the number of StackOverflow questions for R and Python and which tool between R and Python has the greatest predicted growth from 2019 to 2021. The dataset used for analysis was obtained from Kaggle and includes the monthly question counts from December 2011 to December 2019 for 82 data science tags. We began our analysis by exploring the data through several time series visualizations, building univariate SARIMA models for the R and Python series to identify complex patterns of trend and seasonality, and finally building multivariate VAR models in an attempt to answer our research questions. Based on our analysis we found that Python is forecasted to have a 17.97% growth rate from December 2011 to December 2019 while R is forecasted to have a 26.24% growth rate. We also concluded that R has an overall better fit with the data science topics than Python with Classification and Cluster Analysis as significant predictors. We believe that the difference in results between Python and R could be attributed to the fact that R is primarily used for statistical modeling while Python has many other uses, such as software engineering tasks. One major limitation of our analysis is that we chose the variables for our models based on intuition and previous experiences. Thus, there may be better features to predict Python and R question counts. With more time, we would like to extend the time frame of our analysis by including data from 2020 and 2021 to verify the results of our model forecasts. Overall, we believe this analysis provides informative insights on the usage of data science methods in Python and R and how the use of these tools for data science is predicted to grow.

## Introduction

Python and R are two data science tools used frequently in both industry and academia. It would be interesting to know whether these platforms are being used for similar data science topics,

such as regression or classification, or whether people tend to use one over the other for certain topics. It would also be interesting to know which of these tools is expected to dominate over the other in terms of usage for data science. We investigate the use of these programming languages through counts of questions on StackOverflow, a public question-forum that is popular for data scientists and programmers. The main research questions we aim to answer are:

- Which data science topics significantly contributed to the number of StackOverflow questions for Python and R?
- Which tool between R and Python has the greatest predicted growth between 2019 and 2021?

## Data

The data for this analysis was provided by Aishwarya and Vaishnavi V. at TactLabs via the public data warehouse Kaggle (Aishwarya 2020). The data set consists of question counts for various Stack Overflow data science topics. The raw data file has 82 columns of different topics, and 133 rows of monthly question counts from December 2009 to December 2019. Below are the variables used in our analysis:

- python
- r
- regression
- machine_learning
- classification
- cluster_analysis
- time_series

While we were performing the exploratory data analysis, we decided to exclude the first two years (January 2009 to December 2010) from the dataset for a few reasons. To start, some of the topics had no recorded questions in these first two years which negatively affects the modeling process. Similarly, 2009 is around the period when data science and StackOverflow became popular and utilized by people in the field. Consequently, the data in the first two years look drastically different than the remainder of the dataset. Therefore, we decided that it was best to focus on the data from 2011 to 2019. The dataset used in all exploratory data analysis and subsequent modeling contains the monthly counts from December 2011 to December 2019.

## Methods

*Exploratory Data Analysis*

We began our analysis by exploring our data through several time series visualizations. First, each of the relevant time series was plotted on the same graph to understand how each series compared to another and to see if there were any overall patterns. Then, a correlation matrix was created to calculate the strength of the relationships between the series. Monthly plots and decompositions were also made to see if there was any seasonality and where the seasonal components were taking place. Finally, an ACF and PACF plot was made to measure the temporal correlations within each series.

*SARIMA Modeling*

As will be explained later in the results, our exploratory data analysis found that our time series had complex patterns. However, the visualizations from the exploratory data analysis were not sufficient to make modeling conclusions. Consequently, we decided to build an autoregressive integrated moving average model or SARIMA model for the R and Python series to better understand what transformations were needed before fitting a final model. We started the model-building process by selecting SARIMA orders based on the ACF and PACF plots in the exploratory data analysis. Then, we checked the model by examining the residuals, fitted versus observed plots, ACF and PACF plots, QQ plots, and running multiple simulations. After the initial diagnostics, we created an additional model by using the auto.arima() function in R. The parameters for this automatic modeling selection function were decided based on the information gained from the initial model we built. If the model that was created from the auto.arima() function matched the model that was constructed from the exploratory data analysis and the model diagnostics showed no glaring errors, the similar SARIMA model was used to calculate forecasted values. But if the auto.arima() function produced a different model, then the model diagnostics listed earlier were applied again to see if the new model is a better fit to the data. The models with similar diagnostics were compared using AIC, RMSE, and MAE. In the end, the orders from the final SARIMA models are further examined to determine what data transformations are needed before fitting a VAR model.

*VAR Modeling*

While the ARIMA models are helpful in our understanding of the data, they cannot suggest which data science topics are significant to the R and Python series. This is because ARIMA models are used specifically for univariate time series, but our inference question requires understanding the relationships between several time series. Therefore, we fit a vector autoregressive moving average model or VAR model using the VARselect() function in the vars package since it can evaluate the relationship between multiple time series. In the function, we

set the ‘type’ of fitting to none since we manually detrended and deseasonalized the series before fitting the model and set the maximum lag to 13 since any variables past this point would not be constructive to the analysis. This transformation was done by adding an indicator variable for each month to capture the seasonality and modeling the residuals of a simple linear regression for the time series to account for the trends. Again, we performed model checks to see if the selected model from the function was a good fit for the data. The diagnostics included a plot of the fitted versus actuals, ACF and PACFs of the residuals, the forecasts, and the plot of the residuals as well as generating the model summary statistics. If any transformations are required to fit a VAR model, we reversed all alterations in the final forecasts and conclusions. Now that we have a final model, we answered our research questions by analyzing the coefficients and forecasts. Specifically, we checked the model coefficients to determine which data science topics contributed to the R and Python models. Then, we calculated a simple percent growth rate from the final forecasts using the last actual and predicted value to see if R or Python had the highest predicted growth from 2019 to 2021.

## Results

*Exploratory Data Analysis*

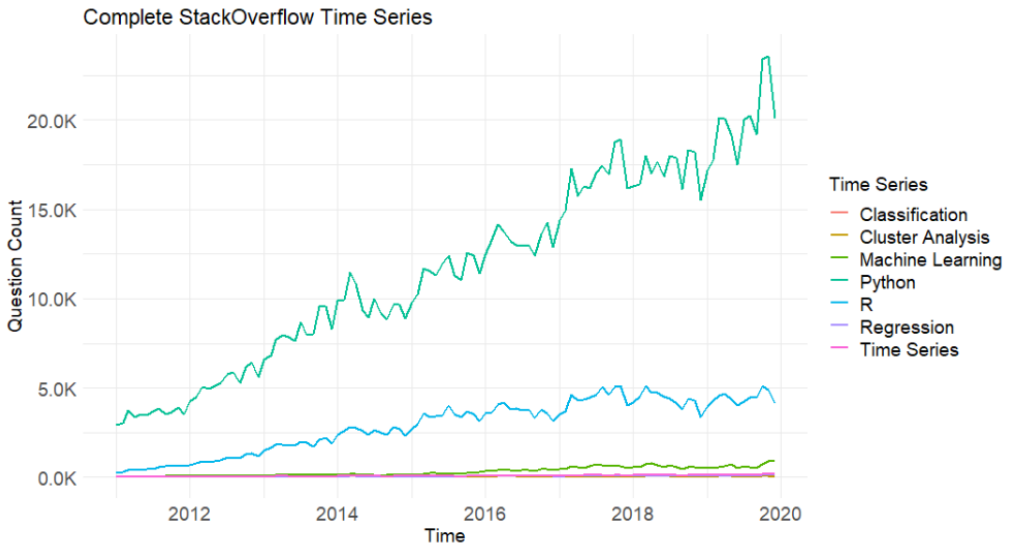


Figure 1: Times series plot of the full data set

Recall that our exploratory data analysis started with a complete view of our time series (Figure 1). Unsurprisingly, Python stands out among all the time series with the largest number of questions since it is arguably the most popular data science tool. R has the second most number of questions, which is expected since R is also favored for academic research but has fewer uses than Python. The remaining topics have significantly fewer monthly questions, with machine

learning having the most questions out of the data science topics. In the individual plots in Figure 2, we can see that every series has an increasing trend which aligns with our general observations in the growing data science job market and demand for these skills. The increasing trend was supported by our correlation plot (Figure 3), which shows that all the series have a strong positive correlation with each other. It is worth noting that R and Python have one of the strongest correlations, which may cause issues with multicollinearity in the modeling process.

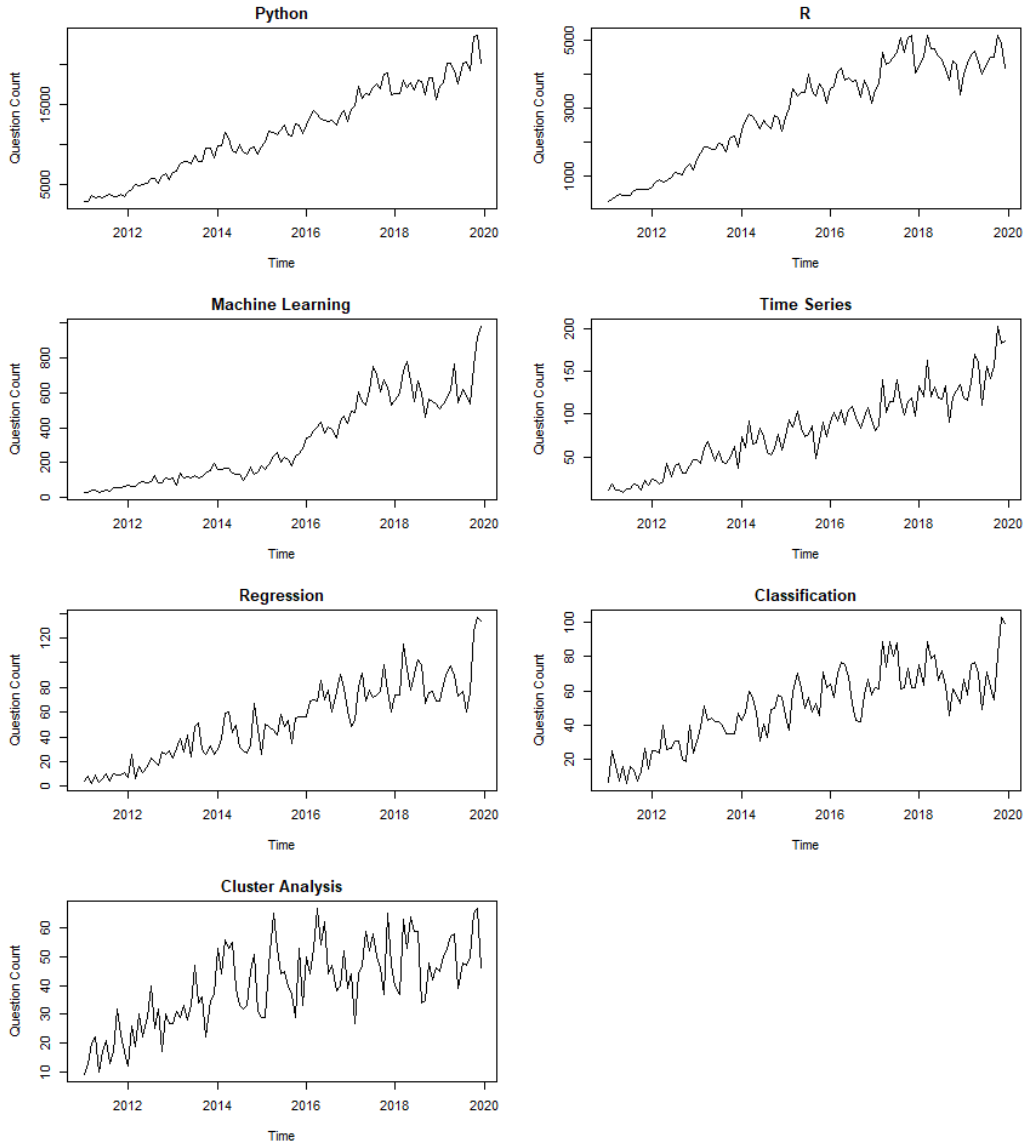


Figure 2: Individual plots for all time series

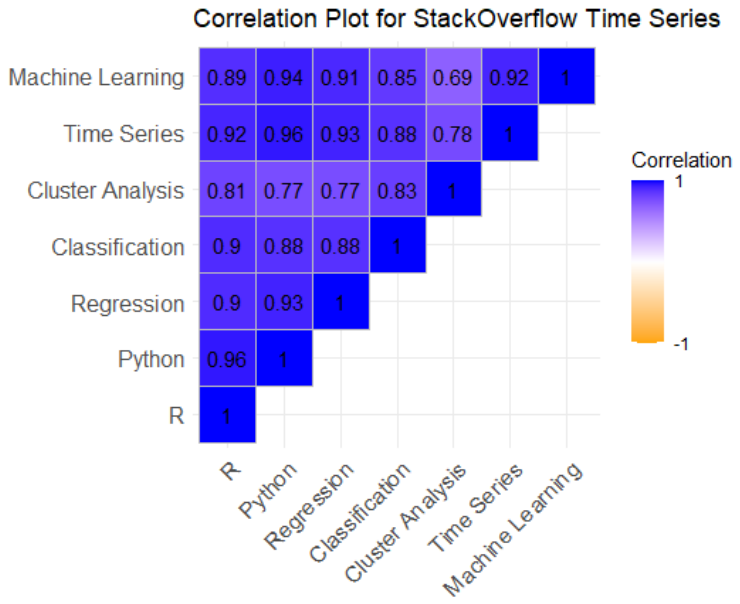


Figure 3: Correlation matrix for all variables of interest

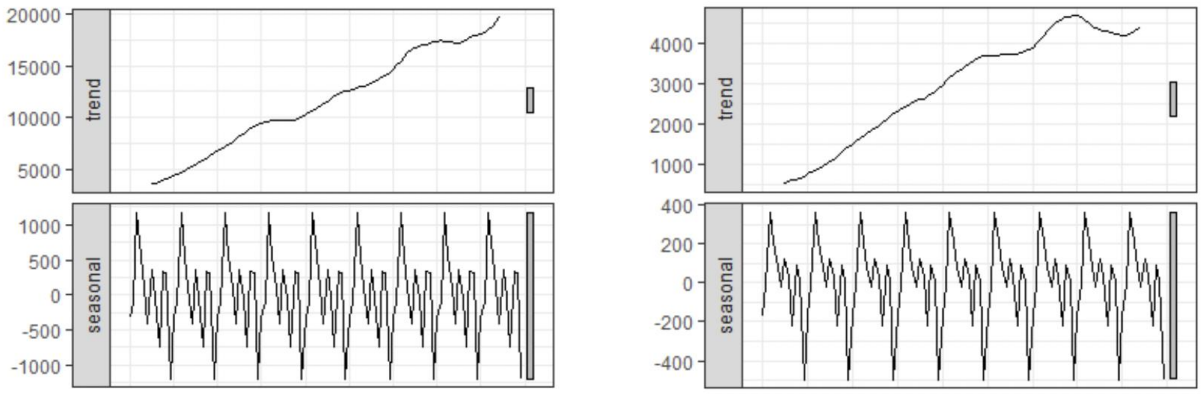


Figure 4: Decomposition of Python (left) and R (right) time series

Additionally, the decomposition of the time series for R and Python shows that there may be seasonality in the time series since there are higher question counts in March, April, and November (Figure 4). The plots also support the idea that a trend is present since the trend component of the decomposition has an increasing linear line. Next, the ACF plots for all the time series show a slow decay with increasing lags, which suggests that there may be an autoregressive behavior to the series (Tech. Apx. page 11). Additionally, the PACF plots occasionally show a small significant correlation around a year-long lag (Tech. Apx. page 12). This supports our observations for the decomposition plots which suggest a seasonality component to the time series.

*SARIMA Modeling*

The first step in our modeling process was to fit a SARIMA model to our Python and R series to learn more about the behaviors we observed in the exploratory data analysis. Recall that the Python process showed autoregressive behavior in the autocorrelation plots, increasing trend, and semi-annual seasonality. Therefore, the first SARIMA model was fitted with a non-seasonal autoregressive term, a seasonal autoregressive term, and a seasonal difference term. Now that we found the first model, we created several diagnostic plots to see if our initial observations translated into a model that fits well with the data. All the diagnostic plots for the R and Python SARIMA models can be viewed in the technical appendix starting on page 13. Firstly, the fitted versus observed plot for Python indicated that the model fits the data well since the predicted values were close to the original values, with a few possible exceptions in 2018 (Figure 5). In some parts of the series, it looks like the model is predicting slightly ahead of the observed values, but overall, it looks to be a good fit for the data.

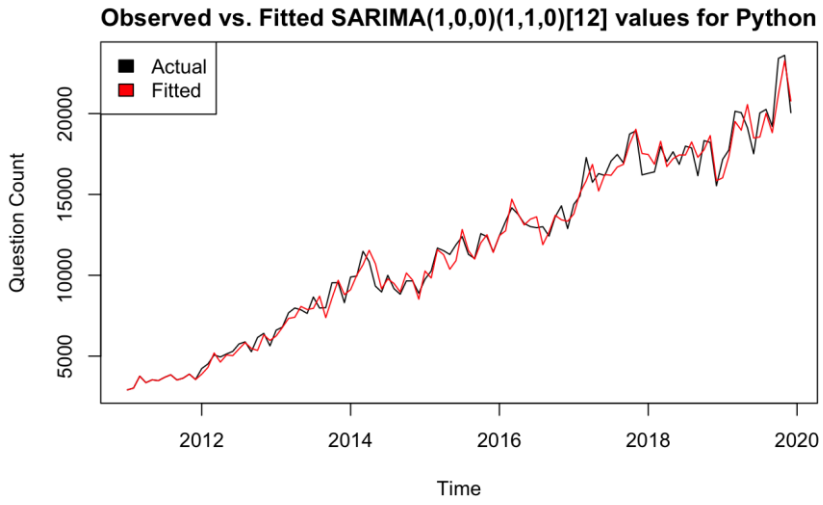


Figure 5: Observed versus fitted plot for Python SARIMA() model

The residuals also appear to be approximately stationary white noise with a constant mean and variance. However, the ACF and PACF plots display a small significant correlation around the 13th lag. These correlations are not significant enough to warrant any action, but it is worth noting that there may be an underlying behavior in the series that the model is not capturing. Also, the QQ plot shows that the residuals are approximately normal. Several of the simulations exhibited similar behaviors to the original StackOverflow time series since they had constant increasing trends with slight annual variations. Lastly, the forecasts from the SARIMA model for Python have an increasing trend with some variations within the year which is what we would expect based on the previous findings (Figure 6). All of these observations suggest that the SARIMA(1,0,0)(1,1,0) model is a good fit to the data and that the original Python time series needs to be detrended and deseasonalized.

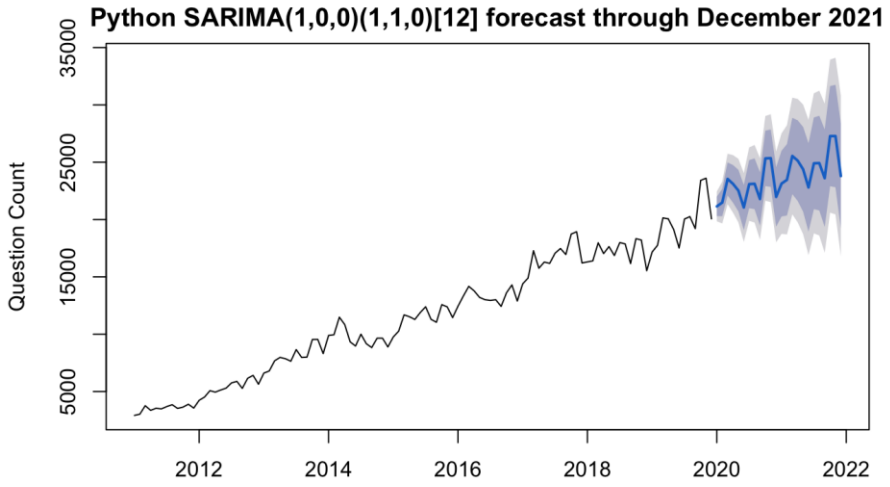


Figure 6: Forecasted Python SARIMA() series through December 2021

The same initial SARIMA(1,0,0)(1,1,0) model was fit to the R time series since the exploratory data analysis showed similar trends and seasonality for both the Python and R time series. Overall, the R time series had similar results for most of the model diagnostic checks except for the ACF and PACF plots. Specifically, the Python time series had some small significant correlations in the ACF and PACF plots while there appear to be no such significant correlations in the R time series.

However, the auto.arima() function in R produced a different model than the one suggested by the exploratory data analysis. Specifically, the auto.arima() function suggests adding a non-seasonal difference term, removing a seasonal autoregressive component, and adding a seasonal moving average term to make a SARIMA(1,1,0)(0,1,1) model. However, when these changes were made, the model diagnostics produced similar results to the initial model created from the exploratory data analysis. Since the diagnostic plots did not suggest a superior model, we looked at the model summary statistics to select a final model. Looking at the summary output on page 30 in the technical appendix, the model that the auto.arima() function picks has slightly lower AIC and RMSE/MAE. Therefore, we selected the SARIMA(1,1,0)(0,1,1) as the best model for the data. The presence of a seasonal and trend effect in the SARIMA model indicates that a transformation to the R series is needed before the VAR modeling process.

The diagnostic plots for the final R model led to similar conclusions about model validity that we discussed previously with the Python model. In the observed versus fitted plot (Figure 7), the model appears to fit the data well since the observed values line up well with the predicted values, despite the predictions being slightly ahead of the observed values in some parts of the series. The residuals appear to be white noise and there are no significant lags in the ACF and

PACF of the residuals. The QQ plot shows the residuals are approximately normal and the simulated plots look like they could reasonably come from the same underlying stochastic process as the observed data. Finally, the forecasts for the R process seem to follow the generally increasing trend with a bit of within-year variability that has been observed in the data so far (Figure 8).

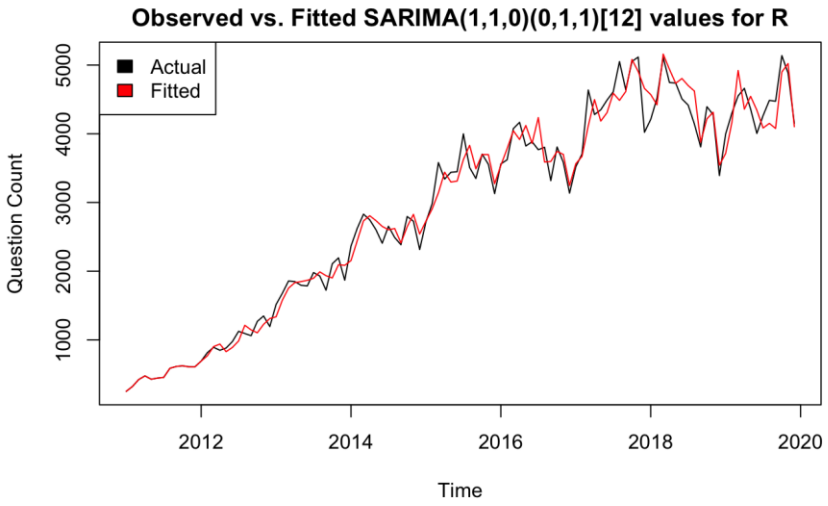


Figure 7: Observed versus fitted plot for R SARIMA() model

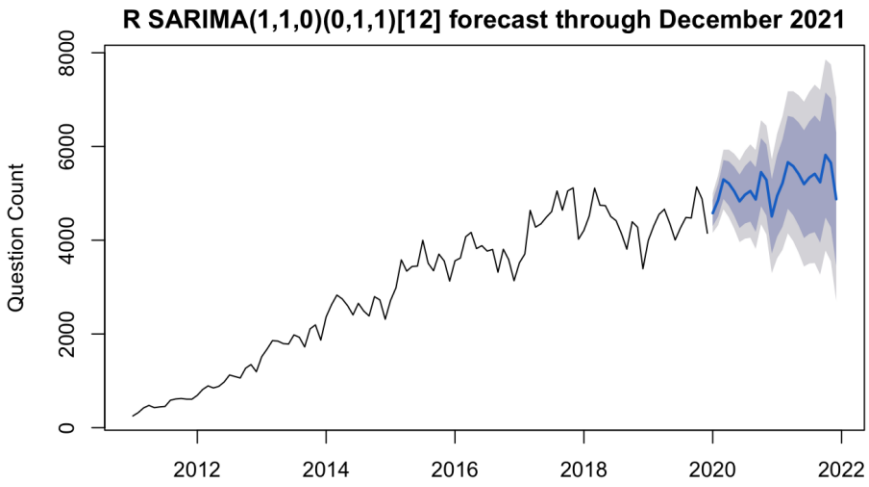


Figure 8: Forecasted R SARIMA() series through December 2021

*VARMA Modeling*

The final SARIMA models and the exploratory data analysis showed that both the R and Python series had a trend and seasonal effect. Therefore, both series were detrended and deseasonalized before fitting a VAR model to answer our research questions about the growth and contributors to the StackOverflow questions. The series were first detrended by linearly regressing the respective time series on their time index and extracting the residuals from these model fits. Another linear regression was then fitted with the detrended series using indicator variables for each month as predictors. Extracting the residuals from this model fit yielded the detrended and deseasonalized data. After transforming the data, we used the VARselect function to fit a VAR model using the data science topics for both the R and Python series. For our Python time series, the automatic model selection found that a VAR(1) model is the best fit for the data (Table 1).

| Variable (Lag 1) | Coefficient Est | Std Error |
|---|---|---|
| Python | 0.67*** | 0.07 |
| Machine Learning | 0.39 | 0.69 |
| Classification | 4.87 | 6.86 |
| Regression | -5.43 | 5.81 |
| Time Series | 0.24 | 4.29 |
| Cluster Analysis | -2.65 | 7.35 |

*** *Indicates a p-value below 0.05*

Table 1: Python VAR(1) model coefficients

Interestingly, the summary statistics of the VAR(1) in Table 1 showed that the model for the detrended and deseasonalized Python series can not be explained by any of the data science topics since all their p-values are above 0.05. The only variable that seems to predict the Python question count is the lagged variable of the Python series itself. The summary also revealed that the R-squared for the model is 0.43 which suggests that the model does not have strong predictive power. In the model diagnostics, the plot of the observed versus fitted values for the Python VAR(1) model showed that the model fit well to the data. However, the predicted values had a worse fit in the data's peaks and troughs (Figure 9).

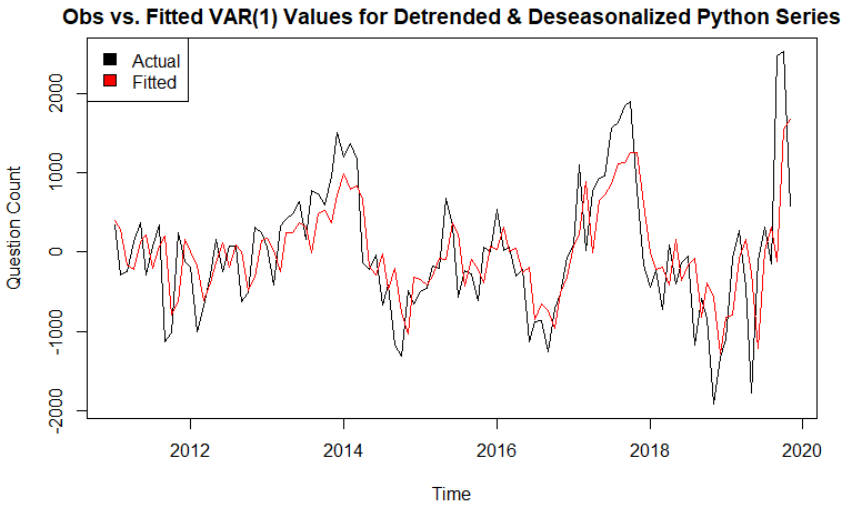


Figure 9: Observed versus expected values for the VAR(1) fitted to the detrended and deseasonalized Python

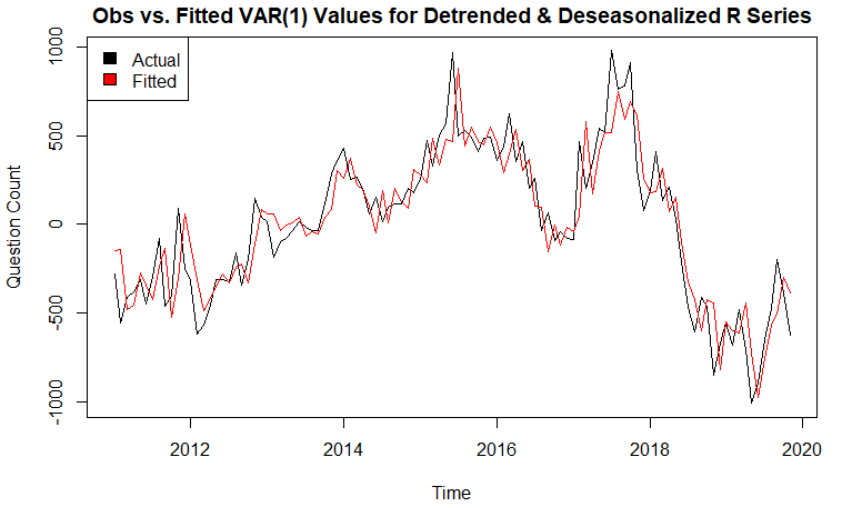


Figure 10: Observed versus expected values for the VAR(1) fitted to the detrended and deseasonalized R time series

Additionally, the ACF and CCF plots occasionally showed significant correlations around the year-long lag. All of these characteristics indicate that the variables selected for the VAR(1) were not constructive and the model may not be the best fit with the data. However, the cause for the poor fit was not obvious from the SARIMA modeling and exploratory data analysis. Therefore, the VAR(1) was kept as the final model. Using the final VAR(1) model, the forecasted values for the Python series were calculated for the next two years (Figure 11). Specifically, Python's predicted number of questions is forecasted to grow from 20,058 in December 2019 to 23,661 in December 2021, a 17.97% growth rate.

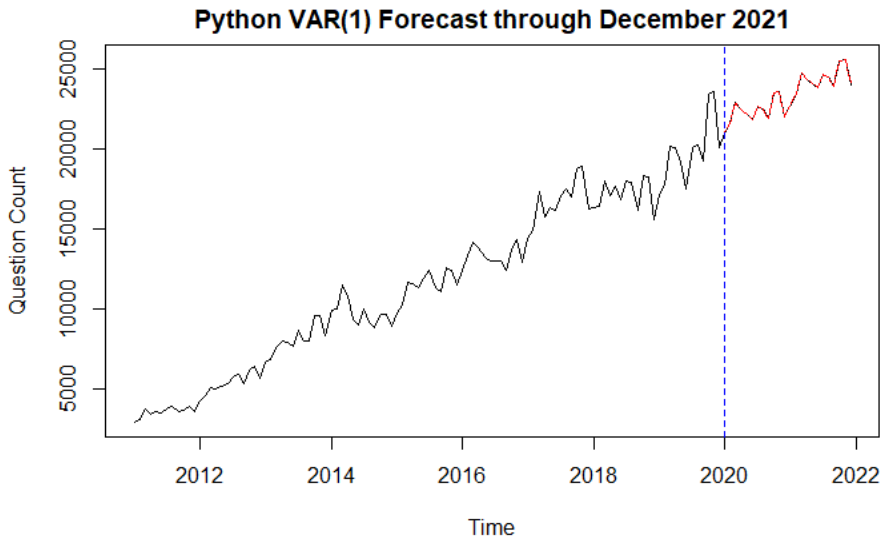


Figure 11: Forecasted Python VAR(1) series through December 2021

| Variable (Lag 1) | Coefficient Est | Std Error |
|---|---|---|
| R | 0.87*** | 0.05 |
| Machine Learning | -0.24 | 0.21 |
| Classification | 7.79*** | 2.15 |
| Regression | -2.57 | 1.78 |
| Time Series | 0.73 | 1.36 |
| Cluster Analysis | -6.18*** | 2.25 |

*** *Indicates a p-value below 0.05*

Table 2: R VAR(1) model coefficients

Moving on to the R time series, the VARselect function determined that a VAR(1) model was also the best fit for the data. Unlike the Python model, the VAR(1) model for R has two variables that significantly contribute to the model: classification and cluster analysis. Both variables have p-values below 0.05 which suggest that they explain a significant amount of variance in the Python series. It is worth noting that the cluster analysis variable has a negative coefficient estimate, which indicates that an increase in the number of cluster analysis questions in the previous month is associated with a decrease in the number of Python questions this month. Additionally, the R-squared value is 0.81 which suggests that the model has strong predictive power. The superior fit is also present in the fitted versus observed plot since the lines are closer

together and have similar values in unusual spikes. The diagnostics for the VAR(1) also indicated a better fit to the data since the residuals for the detrended and deseasonalized R series follow a stationary behavior, with a constant mean and variance. Plus, the ACF and PACF plots also showed fewer instances of significant correlations around the annual lag point. The affirmative evidence from the model diagnostics suggested a good fit for the data, so we selected the VAR(1) model as the final model for the R series. From the model, we calculated that R's predicted number of questions is forecasted to grow from 4,150 in December 2019 to 5,238 in December 2021, a 26.24% growth rate (Figure 12).

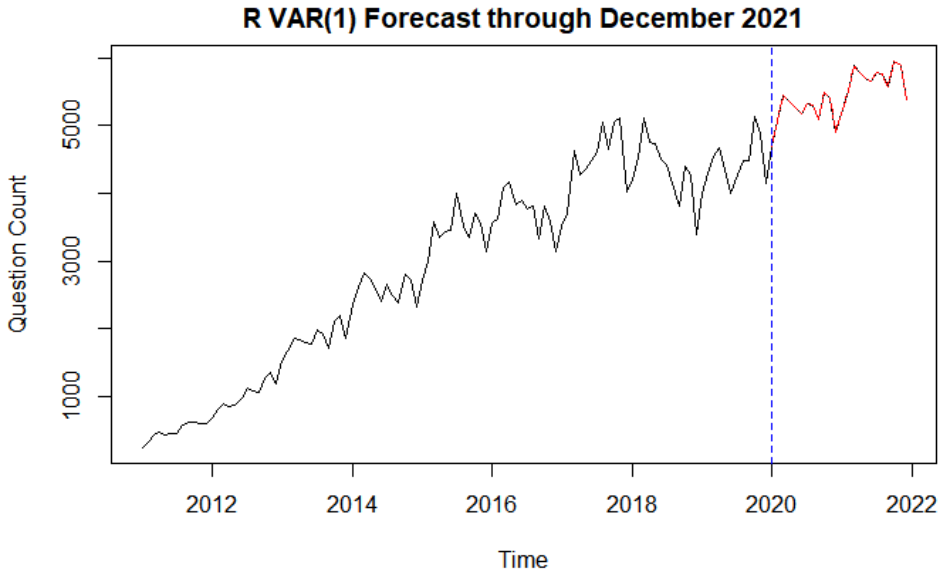


Figure 12: Forecasted R VAR(1) series through December 2021

## Discussion

In this analysis, we built models to predict the number of StackOverflow questions for Python and R using SARIMA modeling. We found that for Python, a model with seasonal and non-seasonal autoregressive terms and a seasonal difference term fits the data well while for R, a model with a non-seasonal autoregressive term, a seasonal moving average term, and a seasonal and non-seasonal difference term fits the data well. We also saw that according to these models, the number of questions is forecasted to continue the generally increasing trend with some within-year variability that has been observed in the data thus far.

We also examined the relationship between different data science topics and the number of StackOverflow questions for Python and R using VAR models. We found that classification and clustering are associated with the number of R questions, and we discovered that none of the data science topics included in modeling are significantly associated with the number of Python

questions. We found that R has an overall better fit with data science topics than Python does. In addition, R is predicted to have a faster growth rate in the number of StackOverflow questions than Python (26.24% vs 17.97%) between the years 2019 and 2021.

We believe that the difference in results we see between Python and R could be attributable to the fact that R is primarily used for statistical modeling while Python has many other uses, such as software engineering tasks. This distinction could help explain the differences we found in significant predictors, adjusted R-squared, and growth rate. We are not sure why clustering and classification are the only significant predictors of R question counts and would be curious to investigate this further in a future analysis.

It is important to note that the series used in this analysis are highly correlated with each other. This correlation may be contributing to the lack of statistically significant predictors we find in the VAR models. Since R and Python are highly correlated with each other, we tried including Python as a predictor in the VAR model for R and vice versa, but these additions did not significantly improve the model fit.

One limitation of this analysis is that we used a limited selection of data science topics to keep the models manageable. We chose the topics based on our intuition and previous experiences, but there may be better features to predict Python and R question counts. Another limitation is feature redundancy in the data. Since posts on StackOverflow can be tagged with multiple topics, it is possible that some questions are being double counted in the data. For example, there is a Python column and a Python 3.0 column, and these tags may both be used on a single question. In addition, we may be losing important or interesting patterns in the data due to monthly aggregation. A future study may examine the counts of questions at a daily or weekly frequency and see if the results are consistent. Finally, it may not be reasonable to compare R and Python since, as previously mentioned, R is primarily used for statistical modeling while Python has several additional uses. This makes it unsurprising that we find that some data science topics are significant predictors of the number of R questions, but not of the number of Python questions.

Some next steps we would take include extending the time frame of our analysis by including data from 2020 and 2021 and verifying the results of our model forecasts with this new data. We would also try including more predictors in the model, such as decision trees and support vector machines, to see if they can predict question counts better than the current data science topics. In addition, we would like to add an indicator variable for whether the observation takes place during the academic semester to account for potential differences in question frequencies when students are in and not in school. Finally, we would consider performing a log transformation on all of the time series before modeling to see if the transformed data produces better model fits.

In summary, this analysis provides informative insights on the usage of data science methods in Python and R and how the use of these tools for data science is predicted to grow. It will be interesting to see whether the model predictions are accurate in determining that R will grow faster than Python in terms of StackOverflow data science topic question counts.

## References

Aishwarya, and Vaishnavi V. (2020), “StackOverflow Questions Count Time Series”, Kaggle, Available at https://www.kaggle.com/datasets/aishu200023/stackindex?select=MLTollsStackOverflow.csv.